# Accessibility

Accessibility This is our belief Design for everyone Carved directly from our Inclusive pillar, this document outlines our unwavering commitment to accessibility. In the Play+ ecosystem, accessibility is not an optional feature or a final checklist item—it is the essential practice of ensuring that our products are perceivable, operable, understandable, and robust for all users, regardless of their abilities, context, or environment. We believe that designing for accessibility leads to a better experience for everyone. By creating clear, flexible, and forgiving interfaces, we empower all users and build a foundation of trust and respect. This guide provides the principles and technical standards necessary to uphold that commitment. Our Standard: WCAG 2.2 AA ■ Play+ adheres to the Web Content Accessibility Guidelines (WCAG) 2.2 at the Level AA conformance standard. All new components and features must meet this minimum standard. While AA is our baseline, we aspire to meet Level AAA criteria where practical, especially concerning color contrast and readability. Core Implementation Areas ■ To ensure our commitment is met in practice, accessibility is broken down into four key areas of implementation. These are the global rules that all components must follow. Visual Accessibility (Perceivable) ■ Ensuring content is visually clear for all users, including those with low vision or color blindness. Guideline Design Rule Developer Implementation **Color Contrast** All text must have a minimum contrast ratio of 4.5:1 against its background. Large text (18pt+ or 14pt+ bold) must have a 3:1 ratio. **Token Safety:** Use pre-validated color token combinations. Automate contrast checks in the CI pipeline using tools like Axe. **No Color-Only Indicators** Never use color as the sole means of conveying information (e.g., an error state). Always supplement with an icon, text, or shape change. **Implementation:** When applying an error color token (e.g., to a border), also add an error icon and link to a descriptive message via aria-describedby. **Clear Focus States** All interactive elements must have a highly visible, consistent focus indicator that does not rely on color alone. **Global Styles:** The global focus ring ($accessibility-focus-ring-*) must be applied to all interactive elements using the :focus-visible pseudo-class. **Text Readability** Typography must be clear and legible. Text must be zoomable up to 200% without breaking the layout. **CSS:** Use rem units for font-size and avoid fixed heights on text containers to allow for resizing. Auditory & Non-Text Accessibility (Perceivable) ■ Providing text-based alternatives for all non-text content. Guideline Design Rule Developer Implementation **Image Alternatives** All informative images must have a clear, concise text description. Decorative images should be ignored by assistive technology. **HTML:** All `<img>` tags must have an alt attribute. For informative images, alt="A description...". For decorative images, alt="". **Icon Alternatives** Icons that convey meaning or action must have a text alternative. **ARIA:** For icon-only buttons, provide an aria-label. For decorative icons within a larger component (like a link with text), use aria-hidden="true". **Video Content** All videos must have accurate, synchronized closed captions. **Implementation:** Use the `<track>` element with a valid WebVTT file for captions.

**Audio Content** A text transcript must be available for all important audio-only content. **Content:** Provide a link to the full transcript on the same page as the audio player. Motor Accessibility (Operable) ■ Ensuring the interface can be fully controlled through various input methods. Guideline Design Rule Developer Implementation **Full Keyboard Navigability** All interactive elements—including links, buttons, inputs, and custom components—must be reachable and operable using only the keyboard. **Semantic HTML:** Use native `<button>`, `<input>`, and `<a>` elements where possible to get keyboard behavior for free. For custom components, use tabindex="0". **No Keyboard Traps** Keyboard focus must never become "trapped" within a component. A user must always be able to navigate into and out of any component. **Focus Management:** For modals or popovers, ensure the Tab key cycles focus within the component, and the Esc key closes it and returns focus to the trigger element. **Sufficient Target Size** All interactive targets must have a minimum size of 44px by 44px, including padding, to be easily used on touch devices. **CSS:** Apply min-width and min-height or sufficient padding to all interactive elements to meet the target size. **Alternative Input** Functionality should not rely exclusively on complex gestures like swiping or dragging. Provide simple click or tap alternatives. **Event Handling:** In addition to drag-and-drop, provide a standard file `<input>` button. For swipe carousels, include "Next/Previous" buttons. Cognitive Accessibility (Understandable) ■ Making the interface clear, predictable, and easy to comprehend. Guideline Design Rule Developer Implementation **Predictable Navigation** Navigation and key interface elements must remain in consistent locations throughout the application. **Layout & Components:** Use shared layout components for headers, sidebars, and page structures to enforce consistency. **Clear Labeling** All form controls and interactive elements must have clear, persistent, and visible labels. **Semantic HTML:** Every `<input>`, `<select>`, and `<textarea>` must have an associated `<label>`. **Actionable Error Messages** Error messages should be written in simple language, explain what went wrong, and suggest a clear solution. **Content Strategy:** Follow a consistent pattern for error messages. Link errors to the relevant field using aria-describedby. **Respect for Motion** Animations should not be distracting or cause physical discomfort. Avoid fast, flashing, or jarring movements. **CSS Media Query:** All animations must be wrapped in a @media (prefers-reduced-motion: no-preference) block. Provide a simpler, non-moving alternative for users who prefer reduced motion. Component Accessibility Requirements ■ The following is a component-by-component breakdown of how the Inclusive pillar is implemented. Text & Heading ■ Contrast: All text and heading colors must pass a 4.5:1 contrast ratio against their background (3:1 for large text). Semantic Structure: Headings must be used in a logical, descending order (H1 -> H2 -> H3) without skipping levels. Readability: Text must use rem units to allow users to scale the font size in their browser settings without breaking the layout. Emphasis: Use  and  tags for semantic emphasis, not <b> or <i> tags which are purely stylistic. Button & Icon Button ■ Keyboard Operability: Must be focusable with Tab and activatable with Enter and Space. Accessible Name: Must have a clear, descriptive text label. Icon-only buttons must have a descriptive aria-label. Focus State: Must display a clear, high-contrast focus ring when focused via keyboard. State Announcement: A

Loading or Processing state must be announced to screen readers, for example by using aria-live regions. Input & Text area ■ Labeling: Every input must have a programmatically associated <label> using the for and id attributes. Error Identification: When validation fails, the input must have aria-invalid="true", and the error message must be linked to the input via aria-describedby. Autocomplete: Use appropriate autocomplete attributes (e.g., autocomplete="email") to help users fill out forms faster and more accurately. Input Instructions: Any formatting requirements or helper text must be programmatically associated with the input. Checkbox & Radio ■ Semantic Grouping: Groups of related radio or checkbox inputs must be wrapped in a <fieldset> with a descriptive <legend> . Keyboard Navigation: Radio groups must be navigable as a single unit using arrow keys. Checkboxes are navigated individually. State Announcement: The checked and unchecked states must be programmatically announced by screen readers. Click Area: The label for each control must be part of the clickable area to provide a larger target. Switch ■ Accessible Role: Must be implemented with role="switch" and use aria-checked to communicate its on/off state. Labeling: Must have a clear, visible label that describes the setting it controls. Keyboard Operability: Must be focusable with Tab and toggleable with the Space key. State Indication: The on/off state must be clear not just from color, but from the visual position of the switch's thumb. Select (Dropdown) ■ Keyboard Navigation: Must be fully operable with the keyboard, including opening the menu (Enter/Space), navigating options (Arrow Keys), selecting an option (Enter), and closing the menu (Esc). State Announcement: The expanded or collapsed state of the menu must be announced using aria-expanded. Focus Management: When the menu is opened, focus must be programmatically moved to the list of options. When closed, focus must return to the trigger button. Accessible Name: The component must be associated with a visible <label> . Image ■ Alternative Text: All informative images must have a descriptive alt attribute. Decorative Images: Images that are purely decorative must have an empty alt attribute (alt="") to be ignored by screen readers. Complex Images: For complex images like charts or graphs, a longer text description must be provided nearby and associated with the image using aria-describedby. Text on Images: Any text placed over an image must have a scrim or solid background to ensure it meets the 4.5:1 contrast ratio. Avatar ■ Accessible Name: The avatar image's alt attribute must contain the user's name (e.g., alt="Jane Doe"). Fallback Contrast: If using initials as a fallback, the text color must have a 4.5:1 contrast ratio against the generated background color. Proximity: The avatar must always be placed in close proximity to a visible text element displaying the user's name. Status Information: User status (e.g., "Online") indicated by a dot should also be available in a text format for screen readers. Badge ■ Accessible Text: If a badge contains only a number (e.g., "3"), it must be given a full accessible name via an aria-label (e.g., aria-label="3 unread notifications"). Contrast: The text inside the badge must have a 4.5:1 contrast ratio with its background. Not Just Color: Badges indicating status (e.g., "Error", "Success") must use both color and text to convey their meaning. Non-Interactive: Badges are for information only and must not be focusable or interactive. Tooltip ■ Triggering: Must appear on both hover and keyboard focus of the trigger element. Programmatic Association: The tooltip's content must be linked to its trigger element using aria-describedby. Not Focusable: The tooltip itself must not be focusable.

Focus remains on the trigger element. Dismissal: Must be dismissible by pressing the Esc key or by moving focus away from the trigger element. Accessibility Tokens ■ Play+ uses design tokens to ensure consistent accessibility experiences. Focus State Tokens ■ --focus-outline-color --focus-outline-style --focus-outline-width These define accessible, visible focus rings for all components. Text Size Tokens ■ --text-base , --text-large , --text-xl Respect user zoom and system settings (use rem/em) Contrast Tokens ■ --color-text-primary-on-bg --color-bg-surface --color-border-active All token pairs must meet WCAG 2.2 AA contrast minimums Token-to-Guideline Mapping ■ Component Token Used Font Button Font/Label 3.75rem / 60px Card Heading Font/Heading 2 3rem / 48px Card Sub Heading Font/Heading 3 2.25rem / 36px Card Body Text Font/Body 1 1.875rem / 30px Input Field Font/Body 1 1.875rem / 30px Testing & Validation Recommendations ■ Play recommends testing for Accessibility through some of the following ways: Automated Testing ■ Integrate into CI/CD: Axe, Lighthouse Validate token safety and focus states Manual Testing ■ Test with screen readers: NVDA, JAWS, VoiceOver Validate all interactions via keyboard navigation Inclusive User Testing ■ Test with assistive tech users (vision, motor, cognitive) Validate across input methods and screen sizes Tools ■ Axe by Deque Lighthouse (Chrome DevTools) NVDA, JAWS, VoiceOver Stark, Color Oracle Axe Lighthouse NVDA JAWS VoiceOver Stark Color Oracle Advanced Accessibility Features ■ These features go beyond compliance and are part of our current implementation strategy: Dark Mode Accessibility : All token combinations must meet contrast standards in both light and dark modes. Live Regions for Dynamic Content : Use aria-live and aria-atomic for modals, toasts, banners. Localized Accessibility : Ensure right-to-left (RTL) compatibility and screen reader accuracy in translated interfaces. Speech Interface Readiness : Design with semantic clarity to support voice assistant integration. Custom Motion Controls : Enable toggling of animation and provide reduced-motion alternatives with prefers-reduced-motion Focus Ring Customization : Users can override focus style via accessibility settings. Accessibility Snapshot Mode : Optional view to visualize all focus zones, alt text, and keyboard paths.